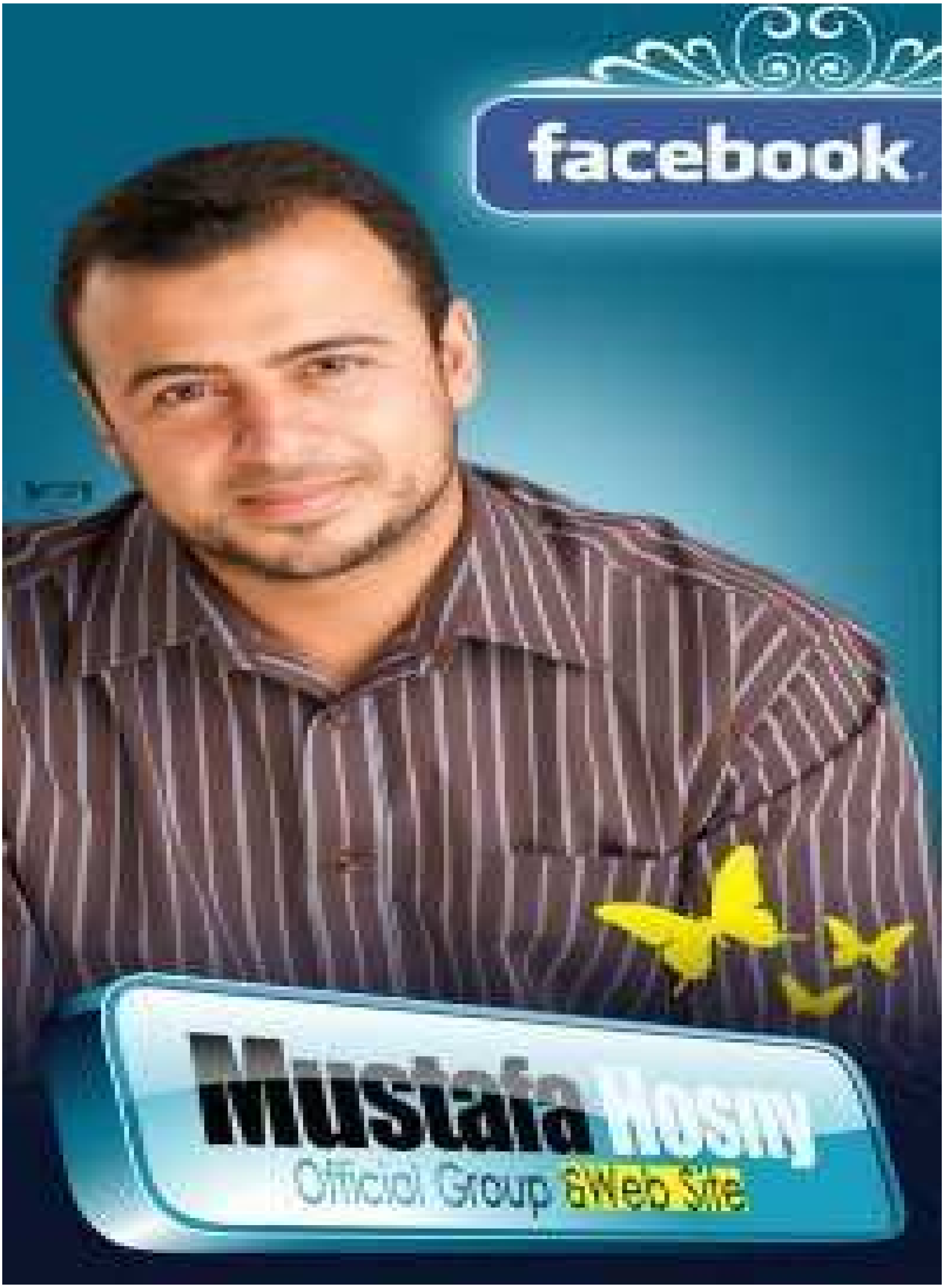

facebook
Mustafa
Official Group &Web Site

الاهداء

الي ذات الحروف الاربعة ....

الي التي اشفقت علي عيني من كثرة القرأءة ......

ها انا ذا اهديك نتاج تلك القرأءة ......

الي والدي العزيز رحمه الله ......

الي اخي العزيز واستاذي محمد ......

الي اخواني الاعزاء معتصم ومصعب وعمار ....

الي اخواتي العزيزات ......

الي جميع اصدقاااائي ..... الي اخي وصديقي عدي سيف الدين ......

الي جميع المسلمين ......

اهدي هذا الكتاااب

عسي ان ينفع الله به المسلمين ......

والله من وراء القصد

بسم الله الرحمن الرحيم

برمجة واجهات المستخدم في لغة الجافا

تأليف وتجميع / أحمد عباس محمد مساعد

قسم نظم الحاسوب والشبكات

السنة الثانية

2010 – 2011 م

المقدمة

في الفصلين الدراسيين السابقين تعرضنا لأساسيات البرمجة بلغة الجافا (الفصل الدراسي الأول) ، وللبرمجة كائنية المنحي object oriented programming(الفصل الدراسي الثاني).

في هذا الكتاب سنتعرض للجزء الاهم من لغة الجافا وهو برمجة واجهات المستخدم . وسنقسم الكتاب الي اربعة اجزاء تصدر علي التوالي ....

المطلوبات :-

- **لدراسة هذا الجزء من الكتاب لابد من دراسة الاصناف المجردة (abstract class) دراسة تامة .**

والله الموفق

احمد عباس محمد مساعد

الفصل الاول

مقدمة الي برمجة واجهات المستخدم

مقدمة الي برمجة واجهات المستخدم

* في كل البرامج السابقة استخدمنا console أو JOptionPane في ادخال البيانات او اخراج المخرجات ، ولكن هذه الطرق بها مشكلة الحدود الضيقة في التعامل .... مثلا اذا اردنا ان نقرأ 20 عدد من المستخدم عن طريق JOptionPane فأننا سنحتاج لإظهار عدد 20 ShowInputDialog وذلك غير فعال في معظم البرامج ويسبب ملل للمستخدم.

* نقوم ببرمجة وجهات المستخدم عن طريق انشاء كائنات (objects) من الاصناف (classes) التالية : JButton , JLabel , JTextFiled , JCheckBox والتي توفر عدد دوال تهيئة (constructors) والتي يمكن إستخدامها لانشاء الكائنات.

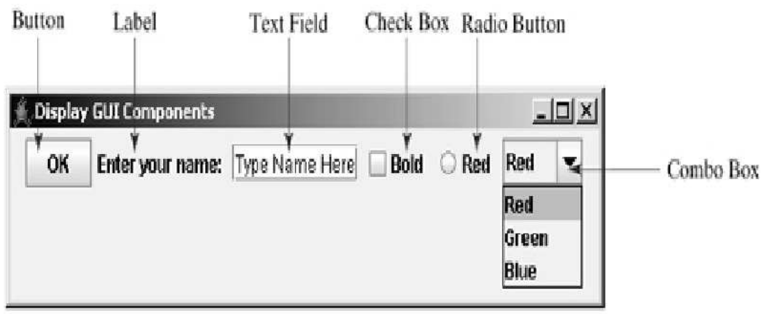
امثلة علي انشاء كائنات من الانواع اعلاه :

```
JButton jb = new JButton("OK");

JLabel jl = new JLabel("Enter Your name here: ");

JCheckBox jbo = new JCheckBox ("BOLD");
```

الشكل الذي تظهر عليه واجهة المستخدم بعد البرمجة :



Swing vs **AWT**

لماذا تحتوي اسماء الكائنات السابقة علي حرف J ؟؟؟ ولماذا لا يكون الاسم من دون هذا الحرف ؟؟

الاجابة علي هذا السؤال ان هنالك اصناف classes من دون حرف J والسبب في ذلك انه في الجافا نستخدم فئتين لبرمجة واجهة المستخدم وهما java.awt و javax.swing .

ملحوظة مهمة :-

سنستخدم كلمة فئة او مكتبة كتعريف لكلمة package الانجليزية في كل فصول الكتاب ، كما سنستخدم كلمة صنف كتعريف لكلمة class ، وكلمة كائن كتعريف لكلمة object .

* عندما قدمت لغة الجافا برمجة واجهات المستخدم لأول مرة كانت كل الاصناف موجودة في مكتبة او فئة تسمي awt وهذه الفئة تقوم بضبط اعدادت البرنامج تلقائيا حسب المنصة التي يتم تشغيل البرنامج عليها ، هذه الاصناف الموجودة في هذه الفئة awt تنفع في بناء واجهات مستخدم بسيطة ولكن لا تجدي نفعا في بناء واجهات مستخدم محترفة ومتميزة.

* مع ظهور النسخة الثانية من الجافا تم استبدال فئة awt بفئة اكثر تميز وكفاءة هي فئة swing والتي تعتمد علي انشاء الكائنات من دون الاعتماد علي المنصة التي يتم تشغيل البرنامج عليها .

* يتم الاشارة الي فئة swing بـlight weight components او المكونات الخفيفة (وذلك لانها لا تعتمد علي منصة التشغيل) ،،،، والاشارة الي فئة awt ب heavy weight components او المكونات الثقيلة (وذلك لانها تعتمد وتتعامل مع منصة التشغيل).

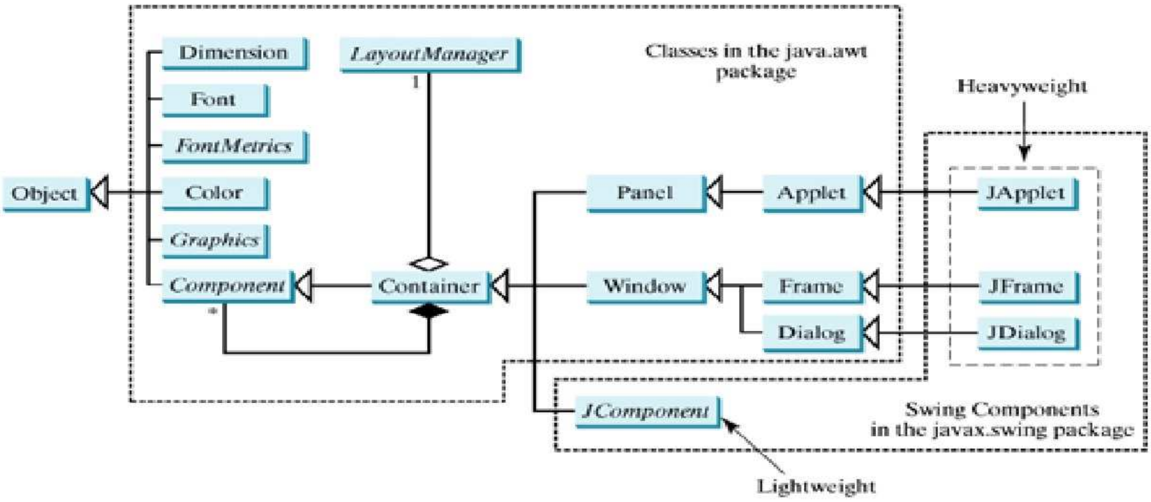
* ومن اجل التفريق بين اصناف فئة awt واصناف فئة swing يتم اضافة اللاحقة J قبل اسم كل صنف من اصناف فئة swing .

* من الافضل التعرد علي البرمجة بواسطة مكتبة او فئة swing (علي الرغم من دعم الجافا لاصناف فئة awt) وذلك لأن هذه الاصناف هي التي ستبقي .

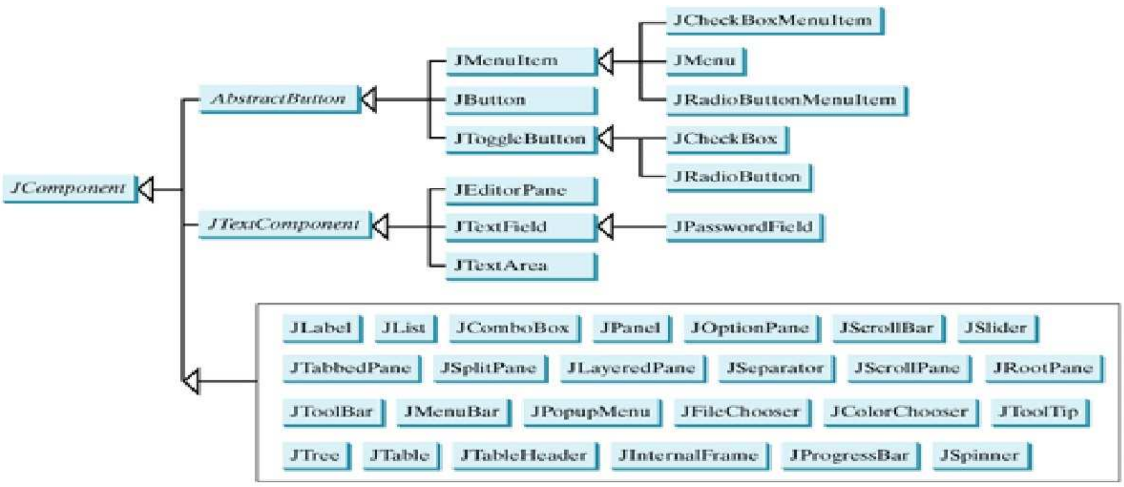
The Java GUI API

GUI Application Program Interface

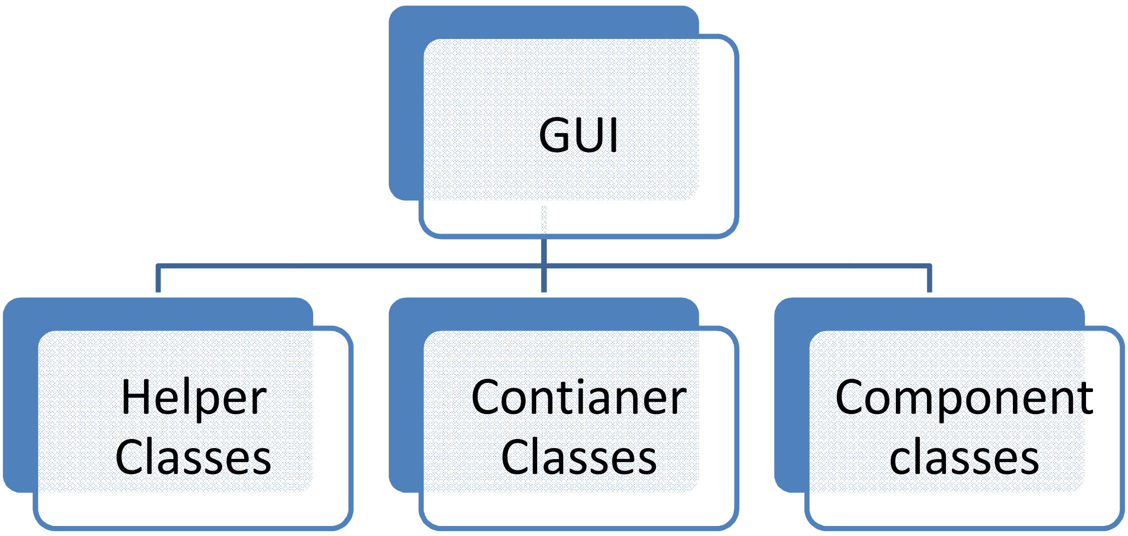
- طريقة تصميم وترتيب الGUI API تعتبر من افضل الامثلة علي استخدام الوراثة والاصناف والواجهات (Interfaces) .



* الرسم اعلاه عبارة عن رسم هيكلي يوضح الاصناف المستخدمة في برمجة واجهات المستخدم. والرسم التالي يوضح اصناف فئة swing



* أصناف البرمجة الرسومية تنقسم الي ثلاثة اقسام:



* container classes او اصناف الحاويات مثل: JFrame , JPanel, JApplet تقوم بأحتواء الكائنات الاخري في داخلها .

* helper classes او الاصناف المساعدة مثل : Color , Graphics, Font,Dimension تستخدم بواسطة اصناف الاحتواء container classes او بواسطة اصناف المكونات component classes وذلك من اجل رسم ووضع المكونات علي الشاشة.

* component classes او اصناف المكونات مثل : JTextFiled , JLabel , JButton فهي عبارة عن اصناف فرعية من الصنف JComponent .

ملحوظة مهمة :-

الاصناف JFrame و JPanel وJDailog و JComponent وكل الاصناف الفرعية منها موجودة في الفئة javax.swing ،،،، اما باقي الاصناف والاصناف الفرعية منها فموجودة في الفئة java.awt ..

Swing Components

- الصنف component هو الأب لكل الاصناف المتستخدمة في برمجة واجهة المستخدم في لغة الجافا .... بينما الصنف JComponent هو الاب لكل الاصناف الخفيفة (swing classes) .
- الصنف JComponent عبارة عن صنف مجرد ولايمكن انشاء كائنات منه ولكن بدلا من ذلك انشاء الكائنات من الاصناف الفرعية منه مثل JFrame .

```
JButton But = new JButton ("OK");
```

Container Classes

* اصناف المحتوي هي عبارة عن اصناف تستخدم لاحتواء المكونات ،، Window , Panel , Applet , Frame , Dialog هي اصناف الاحتواء الخاصة بمكتبة awt.

* بالنسبة لمكتبة swing فأن اصناف الاحتواء هي Component , Container , JFrame , JDialog , JApplet , JPanel.

تفصيل الاصناف :-

* Container تستخدم لتجميع المكونات واللوحات والاطارات وتعتبر الابلت من امثلة هذه الاصناف.

* JFrame هو عبارة عن نافذة غير محتواة في نافذة اخري ويستخدم لاحتواء بقية مكونات واجهة المستخدم في الجافا.

* JDialog هو عبارة عن نافذة منبثقة او صندوق رسالة يستخدم بصورة مؤقتة لادخال معلومات من المستخدم او لعرض معلومات اضافية للمستخدم

* JApplet هو عبارة عن صنف فرعي من الصنف Applet ولابد من عمل امتداد منه للتعامل معه .

* JPanel هو عبارة عن حاوية غير مرئية تستخدم لاحتواء مكونات البرمجة ويمكن ان يكون متداخلا ، ويمكن اضافة لوحة دخل لوحة اخري ،،، ويمكن استخدامها لرسم بعض الاشكال.

Helper Classes

* الاصناف المساعدة مثل Graphics , Color , Font ليست اصناف فرعية من الصنف Component وانما تستخدم لتنسيق والتحكم في مظهر الواجهات.

* الصنف Graphics عبارة عن صنف مجرد ويستخدم في رسم الاشكال البسيطة.

* الصنف Color يستخدم في التحكم في الوان جميع مكونات الواجهات.

* الصنف Font يستخدم في تحديد نوع وحجم النص في الواجهات المختلفة .

* الصنف FontMetrics يستخدم في الحصول علي خصائص الالوان الخاصة بمكونات الواجهه.

* الصنف Dimension يستخدم في التحكم في طول وعرض مكونات الواجهه.

* الصنف LayoutManager هو عبارة عن interface يحدد الكائن المنشي منه طريقة احتواء المكونات في الحاويات.

ملحوظة :-

الاصناف المساعدة موجودة في فئة java.awt وذلك لان فئة swing لم تستبدل كل اصناف هذه الفئة وانما فقط اصناف المكونات وابقت علي باقي الاصناف .

الاطارات Frames

* لكي نتمكن من تصميم واجهة مستخدم لابد من انشاء واحد من اثنين اما إطار او أبلت وذلك لاحتواء مكونات الواجهه وفي هذا الكتاب سنتعرض للتعامل مع الإطارات فقط .

Creating a frame :-

لكي نقوم بأنشاء اطار نستخدم الصنف JFrame والذي نفصل تكوينه في الشكل التالي :

+JFrame // constructor
+JFrame (String Title) // cons with title
+setSize(int width , int height) // void // to set the size of f
+setLocation(int x, int y) // set the location of the frame
+setVisible(Boolean visi) // make the frame visible or not
+setDefaultCloseOperation(int co) // what happ on close
+setLocationRelativeTo(component c) // void connect component together

مثال 1:

انشاء اطار عرضه 400 بكسل وطوله 300 بكسل وغير مرتبط باي مكون اخر وعند اغلاقه يغلق البرنامج فورا ،، وعنوانه SUST Scoring System ؟؟؟

الحل

هذا هو الكود الخاص بحل المثال

```
import javax.swing.*;

public class MyFrame {
  public static void main(String[] args) {
    JFrame frame = new JFrame("SUST Scoring System"); // Create a frame
    frame.setSize(400, 300); // Set the frame size
    frame.setLocationRelativeTo(null); // New since JDK 1.4
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true); // Display the frame
  }
}
```

شرح المثال :

في السطر الاول نقوم بعمل استيراد للمكتبة او الفئة javax.swing وعلامة النجمة في العبارة تدل علي اننا نريد استيراد كل الاصناف الموجودة في المكتبة.

في السطر الخامس نقوم بإنشاء كائن من الصنف JFrame ونرسل له القيمة الخاصة بالعنوان والتي هي SUST Scoring System

في السطر السادس نقوم بتحديد ابعاد الاطار ونرسل لها اولا العرض وهو 400 ومن ثم نرسل لها الطول وهو 300

في السطر السابع سنقوم بتحديد ما إذا كان هذا الاطار مرتبط بإطار او مكون اخر من المكونات

في السطر الثامن نحدد العملية التي تحدث عند اغلاق الاطار

في السطر التاسع نحدد ظهور الاطار من عدمه ونرسل له القيمة true لإظهار الاطار و false لإخفاء الاطار .

* المثال السابق يوضح لنا كيفية انشاء إطار ،، ولكن ليس هناك معني لانشاء اطار فارغ لذلك سانضيف مكون جديد عبارة عن زر للموافقة OK button

البرنامج بعد التعديل -:

```java
import javax.swing.*;

public class MyFrameWithComponents {
  public static void main(String[] args) {
    JFrame frame = new JFrame("MyFrameWithComponents");

    // Add a button into the frame
    JButton jbtOK = new JButton("OK");
    frame.add(jbtOK);

    frame.setSize(400, 300);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setLocationRelativeTo(null); // Center the frame
    frame.setVisible(true);
  }
}
```

شرح التعديل :

بعد انشاء الاطار وقبل تحديد الخصائص الخاصة بالاطار نقوم بإنشاء كائن من الصنف JButton ونرسل له النص الذي سيظهر في الزر

ومن ثم نقوم باضافة الزر الي الاطار عن طريق استخدام الدالة add وذلك كالاتي :

Frame_name.add(Button name);

حيث ان :

Frame _name == اسم الاطار

Button name == اسم الزر

ملحوظة :-

لابد من التفريق بين اسم الزر والنص الذي يظهر علي الزر ،، فأسم الزر هو اسم الكائن الذي ننشئه ،، اما النص الذي يظهر علي الزر فهو النص الذي نرسله الي دالة التهئية عند انشاء الكائن.

مدير التخطيط Layout Manager

في اغلب البرامج المرئية يتم وضع المكونات عن طريق برمجة معقدة جدا ومتعلقة بالبكسلات حيث يتم تحديد موقع كل مكون بدقة تامة
ولكن هذا النظام له عيب انه قد لاينفع في بعض الاجهزة.
* يوفر مدير التخطيط طريقة سهلة لوضع المكونات في الواجهه.

ملحوظة :- توفر الجافا وتدعم وضع المكونات في الواجهه عن طريق البرمجة المتعلقة بالبكسل ولكنها غير مستخدمة بصورة كبيرة .

في لغة الجافا يتم وضع المكونات في حاويات وفي المثال السابق لم نحدد اين يتم وضع الزر في داخل الاطار ، ولكن مدير التخطيط قام بوضعه في المكان المناسب
يتم الصنف LayoutManager واي صنف من هذا النوع يجب ان يعمل اعادة هيكلة للواجهة LayoutManager.

```
LayoutManager layoutManager = new XLayout();
container.setLayout(layoutManager);
```

حيث x عبارة عن متغير حسب نوع دير التخطيط
ونقوم باضافة مدير التخطيط الي الحاوية عن طريق العبارة الثانية .

Flow layout

مدير التدفق او flow layout هو ابسط نوع من مدراء التخطيط وفيه يتم ترتيب المكونات من الشمال الي اليمين وبنفس الترتيب الذي تمت اضافتها به ،، وعندما يمتلئ سطر يتم البدء والمواصلة في سطر جديد ويمكن تحديد المكان الذي يتم وضع المكونات فيه عن طريق استعمال احدي هذه الثوابت الثلاثة FlowLayout.LEFT و FlowLayout.RIGHT و FlowLayout.CENTER.

* جدول يوضح التفصيل للصنف FlowLayout

- Int alignment \\ center or left or right
- Int hgap \\ horizontal gap
- Int vgap \\ vertical gap
+FlowLayout() \\ constructor
+FlowLayout(int all) \\ cons with alignment
+FlowLayout(int all, int vgap , int hgap) \\ cons with ali and vgap and hg

مثال :-

برنامج يقوم بإضافة ثلاثة لافتات وثلاثة مربعات نص في داخل إطار وتنظيمها عن طريق FlowLayout ،، سيتغير ترتيب المكونات اذا غيرت الحجم

الحل

الكود الخاص بالبرنامج :

```
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.JFrame;
import java.awt.FlowLayout;

public class ShowFlowLayout extends JFrame {
  public ShowFlowLayout() {
    // Set FlowLayout, aligned left with horizontal gap 10
    // and vertical gap 20 between components
    setLayout (new FlowLayout(FlowLayout.LEFT, 10, 20));

    // Add labels and text fields to the frame
    add(new JLabel("First Name"));
    add(new JTextField(8));
    add(new JLabel("MI"));
    add(new JTextField(1));
    add(new JLabel("Last Name"));
    add(new JTextField(8));
  }

  /** Main method */
  public static void main(String[] args) {
    ShowFlowLayout frame = new ShowFlowLayout();
    frame.setTitle("ShowFlowLayout");
    frame.setLocationRelativeTo(null); // Center the frame
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(200, 200);
    frame.setVisible(true);
  }
}
```

* البرنامج السابق يقوم بانشاء برنامج بطريقة مختلفة عن المثال الذي قبله .

الطريقة المفضلة في انشاء الواجهات الرسومية :-

1- انشاء اطار في دالة التهيئة وجعله ممتد من الصنف JFrame .

2- يمكن جعل الصنف ممتد اذا احتجنا لاستخدامه في مكان اخر.

3- اعادة استخدام الصنف الذي كتبناه.

* من هذه النقطة سنقوم بجعل جميع الاصناف المستخدمة امتداد للصنف JFrame ونقوم بانشاء وبرمجة الواجهه في دالة التهيئة الخاصة بالصنف ،، ويتم استدعاء الدالة في الدالة الاساسية main method.

* اذا استبدلنا الارقام في السطر رقم 10 بالارقام 0 و 0 فسوف يتم وضع المكونات خلف بعضها البعض من دون فراغات.

* في نفس السطر السابق قمنا بإنشاء كائن عشوائي من الصنف FlowLayout.

* الدالة setTitle تستخدم لوضع عنوان للاطار.

ملحوظة مهمة :

لاتنسي استخدام الكلمة المحجوزة new عند انشاء كائن من الصنف FlowLayout واستخدام الدالة setLayout كالتالي

```
setLayout(new FlowLayout());
```

Grid Layout

* هذا النوع من مدراء التخطيط يقوم بترتيب المكونات في شكل مصفوفة من الاعمدة والصفوف ، ويتم فيه ترتيب المكونات من الايسر الي الايمن والبدء بالسطر الاول والذي يليه وهكذا ، ويتم ترتيب المكونات حسب الترتيب الذي تمت اضافتها به .

توضيح المتغيرات والدوال في صنف grid layout :

- Int rows \\ number of rows in the frame
- Int columns \\ number of columns in the frame
- Int hgap \\ the horizontal gap
- Int vgap \\ the vertical gap
+ GridLayout() \\ empty constructor
+ GridLayout(int rows, int columns) \\ constructor with rows and colms
+ GridLayout(int rows, int columns , int hgap , int vgap) \\ with all const

ويمكن تحديد عدد الصفوف والاعمدة عند انشاء كائن من الصنف grid layout وهنالك بعض القواعد الاساسية في تنظيم الصفوف والاعمدة :

1- عدد الاعمدة او عدد الصفوف يمكن ان يكون صفر ولكن لا يمكن ان يكون كل من العددين صفرا. اذا كان واحدا صفر فالاخر لابد ان يكون رقم موجب غير الصفر . البعد الغير صفري يتم تثبيت الرقم الذي تم ادخاله اما البعد الصفري فيتغير بالزيادة عند زيادة مكون وبالنقصان عن نقص مكون .

مثلا :

اذا عرفنا كائن من الصنف grid layout يحتوي علي ثلاثة اعمدة وصفر عمود ،، وكان لدينا 10 مكونات لوضعها في الاطار فسنجد ان الاطار قد قسم الي اربعة صفوف وثلاثة اعمدة وفي كل صف 3 مكونات عد الاخير فانه يكون فيه مكون واحد.

2- في حالة ادخال البعدين كا ارقام غير صفرية يتم تثبيت عدد الصفوف ويقوم المخطط بحساب عدد الاعمدة تلقائيا.

مثال :-
برنامج يستخدم grid layout لانشاء اطار به 3 لافتات و3 مربعات نص
الحل
هذا البرنامج هو نفس البرنامج السابق مع تغيير المخطط من flow layout الي grid layout

```java
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.JFrame;
import java.awt.GridLayout;

public class ShowGridLayout extends JFrame {
  public ShowGridLayout() {
    // Set GridLayout, 3 rows, 2 columns, and gaps 5 between
    // components horizontally and vertically
    setLayout(new GridLayout(3, 2, 5, 5));

    // Add labels and text fields to the frame
    add(new JLabel("First Name"));
    add(new JTextField(8));
    add(new JLabel("MI"));
    add(new JTextField(1));
    add(new JLabel("Last Name"));
    add(new JTextField(8));
  }

  /** Main method */
  public static void main(String[] args) {
    ShowGridLayout frame = new ShowGridLayout();
    frame.setTitle("ShowGridLayout");
    frame.setLocationRelativeTo(null); // Center the frame
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(200, 125);
    frame.setVisible(true);
  }
}
```

* عند تغيير حجم الاطار لا يتغير ترتيب المكونات كما كان يحدث في grid layout وانما تكون كل القيمة ثابتة بما فيها الهوامش و الفراغات بين المكونات.
* في كل من مخطط FlowLayout ومخطط GridLayout ترتيب اضافة المكونات مهم جدا لانه هو الذي يحدد ترتيب المكونات في الاطار او ال frame .

Border Layout

* هذا النوع من مدراء التخطيط يقوم بتقسيم الشاشة الي خمسة اقسام east , west , north , south , center

* يتم اضافة المكونات عن طريق استخدام الدالة add مع تحديد مكان المكون كالتالي :

Add(component , index);

* حيث ان index هو عبارة عن ثابت واحد من خمسة ثوابت هي :

BorderLayout.SOUTH , BroderLayout.CENTER , BorderLayout.NORTH , BorderLayout.WEST , BorderLayout.EAST .

* تفصيل الدوال والمتغيرات في صنف Border Layout : -

- Int hgap
- Int vgap
+ BorderLayout()
+ BorderLayout(int hgap , int vgap)

* المكونات التي يتم اضافتها في شمال وجنوب الاطار يتم عمل تمديد ليها علي امتداد الاطار.

مثال :

برنامج يقوم بإضافة 5 ازار بعناوينها حسب الاتجاهات

الحل

كود البرنامج :

```java
import javax.swing.JButton;
import javax.swing.JFrame;
import java.awt.BorderLayout;

public class ShowBorderLayout extends JFrame {
  public ShowBorderLayout() {
    // Set BorderLayout with horizontal gap 5 and vertical gap 10
    setLayout(new BorderLayout(5, 10));

    // Add buttons to the frame
    add(new JButton("East"), BorderLayout.EAST);
    add(new JButton("South"), BorderLayout.SOUTH);
    add(new JButton("West"), BorderLayout.WEST);
    add(new JButton("North"), BorderLayout.NORTH);
    add(new JButton("Center"), BorderLayout.CENTER);
  }

  /** Main method */
  public static void main(String[] args) {
    ShowBorderLayout frame = new ShowBorderLayout();
    frame.setTitle("ShowBorderLayout");
    frame.setLocationRelativeTo(null); // Center the frame
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(300, 200);
```

```
        frame.setVisible(true);
    }
}
```

* هذا البرنامج هو نفس البرنامج السابق مع إستبدال المخطط grid layout بالمخطط border layout .

ملحوظة مهمة :-
لاحظ الفرق بين دالة add ل grid layout ول flow layout وهي كالتالي :

Add(component)

وبين دالة add ل border layout وهي كالتالي :

Add(component , index)

* الحالة الافتراضية في border layout هي الوسط اي ان

Add(component)
Is equal vent to ==
Add(component , BorderLayout.CENTER);

خصائص مدراء التخطيط

مدراء التخطيط لديهم خصائص تتغير ديناميكيا وهي كالتالي :

FlowLayout:

- alignment : وهي لتحديد المكان .
- vgap : وهي لتحديد الهامش الافقي
- hgap : وهي لتحديد الهامش العمودي

للتعامل مع الخصائص اعلاه نستعمل الدوال التالية

setAlignment
setVgap
setHgap

GridLayout :

- rows لعدد الصفوف
- columns لعدد الاعمدة
- vgap , hgap : لتحديد الهوامش الافقية والعمودية

BorderLayout :

- vgap , hgap : لتحديد الهوامش الافقية والعمودية .

The color class
صنف اللون

* يمكن التحكم في الوان المكونات باستخدام الصنف java.awt.color وبما ان الالوان هي عبارة عن مزيج من الالوان الاساسية الثلاثة الاحمر والاخضر والازرق والتي يتم تمثيل كل واحد منها بقيمة من دون اشارة تدل علي قوتها وتتدرج القيمة من 0 للالوان الغامقة الي 255 للالوان الخفيفة . هذا النظام يعرف بنظام RGB

* يمكن انشاء لون جديد بإستخدام دالة تهيئة كالتالي :

```
public Color(int r, int g, int b);
```

حيث تشير الاحرف r,g,b الي شدة الالوان المكونة للون المطلوب .... كمثال :

```
Color color = new Color(128, 100, 100);
```

ملحوظة مهمة :-
المتغيرات الخاصة بشدة الالوان تتدرج قيمتها من 0 الي 255 وعند تمرير قيمة خارج هذا الحد program will throw the following exception :

```
IllegalArgumentException
```

* يمكن استخدام الدالة setBackground(Color c) والدالة setForeground(Color c) لتحديد لون خلفية المكون المعني ولون المقدمة

مثال :-
تحديد لون الخلفية والمقدمة لزر في واجهة رسومية ؟؟؟

الحل

```
JButton jbtOK = new JButton();
jbtOK.setBackground(color);
jbtOK.setForeground(new Color(100, 1, 1));
```

* يمكن استخدام واحد من الالوان التالية بدلا من كتابة الارقام والالوان هي :
(black, blue, cyan, darkGray, gray, green, lightGray, magenta, orange, pink, red, white, yellow)

مثال :-
نفس المثال السابق عن طريق استخدام الالوان بدلا من الارقاااام؟؟؟

الحل

فقط نغير في السطر الثالث ليصبح كالتالي :

```
jbtOK.setForeground(Color.red);
```

Font Class

صنف اللون

* يمكن انشاء خط عن طريق الصنف java.awt.Font ويمكن التحكم في خطوط المكونات عن طريقة الدالة setFont .

دالة التهيئة لصنف اللون هي كالتالي :

```
public Font(String name, int style, int size);
```

* قيمة المتغير name عبارة عن واحد من القيم التالية : SansSerif, Serif, Monospaced, Dialog, or DialogInput .

* قيمة المتغير style عبارة عن واحد من القيم التالية : `Font.PLAIN` (0), `Font.BOLD` (1), `Font.ITALIC` (2), and `Font.BOLD+Font.ITALIC` (3) .

* اما قيمة المتغير size فهي عبارة عن اي رقم موجب .

مثال :-

انشاء خطين جديدين باستخدام الصنف java.awt.Font واستخدام احدهما في زر في الواجهة

الحل

```
Font font1 = new Font("SansSerif", Font.BOLD, 16);
Font font2 = new Font("Serif", Font.BOLD + Font.ITALIC, 12);

JButton jbtOK = new JButton("OK");
jbtOK.setFont(font1);
```

ملحوظة مهمة :-

اذا كان النظام الخاص بالجهاز يدعم انواع من الخطوط غير المذكورة اعلاه فيمكن استخدامها في البرنامج عن طريق انشاء كائن وكتابة نفس اسم الخط المطلوب في خانة الاسم .

إستخدام اللوحات كحاويات فرعية

* لكي نفهم اهمية هذا الموضوع علينا ان نتخيل اننا نريد وضع 10 ازرار ومربع نص في اطار ،، فسيتم وضع الازرار في ترتيب صفوف واعمدة حسب grid layout اما مربع النص فسيتم وضعه في صف منفصل تماما مما يؤثر علي الشكل الجمالي للبرنامج .

* من الصعب الحصول علي الشكل المطلوب للبرنامج عند وضع جميع المكونات في حاوية واااحدة فقط .

* في برمجة الواجهات الرسومية في الجافا تتيح اللغة ميزة تقسيم الاطار الي عدد من اللوحات تعتبر كل لوحة منها كحاوية فرعية.

- فنقوم كحل للمشكلة اعلاه باضافة الازرار العشرة الي لوحة ومن ثم نضيف اللوحة الي الاطار .

* يمكن انشاء لوحة عن طريق دالة التهيئة التالية :

```
JPanel p = new JPanel();
new JPanel(LayoutManager x)
```

حيث x: عبارة عن نوع من مدراء التخطيط (ارجع لدرس مدراء التخطيط).

مثال :-

انشاء لوحة اسمها p ومن ثم اضافة زر اليها ومن ثم اضافة اللوحة الي الاطار f ؟

الحل

```
JPanel p = new JPanel();
p.add(new JButton("OK"));
f.add(p);
```

مثال :-

انشاء واجهة رسومية لبرنامج يحاكي عمل الميكروييف ؟

الحل



* شكل الواجهة الرسومية المطلوب برمجتها .

الكود الخاص بالبرمجة :-

```
1  import java.awt.*;
2  import javax.swing.*;
3
4  public class TestPanels extends JFrame {
5    public TestPanels() {
6    // Create panel p1 for the buttons and set GridLayout
7    JPanel p1 = new JPanel();
8    p1.setLayout(new GridLayout(4, 3));
9
10   // Add buttons to the panel
11   for (int i = 1; i <= 9; i++) {
12     p1.add(new JButton("" + i));
13   }
14
15   p1.add(new JButton("" + 0));
16   p1.add(new JButton("Start"));
17   p1.add(new JButton("Stop"));
18
19   // Create panel p2 to hold a text field and p1
20   JPanel p2 = new JPanel(new BorderLayout());
21   p2.add (new JTextField("Time to be displayed here"),
22     BorderLayout.NORTH);
23   p2.add (p1, BorderLayout.CENTER);
24
25   // add contents into the frame
26   add(p2, BorderLayout.EAST);
27   add(new JButton("Food to be placed here"),BorderLayout.CENTER);
29   }
30

31   /** Main method */
32   public static void main(String[] args) {
33     TestPanels frame = new TestPanels();
34     frame.setTitle("The Front View of a Microwave Oven");
35     frame.setLocationRelativeTo(null); // Center the frame
36     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
37     frame.setSize(400, 250);
38     frame.setVisible(true);
39   }
40 }
```

* الدالة setLayout معرفة اصلا في الصنف java.awt.Container وبما ان الصنف Panel صنف فرعي من الصنف Container فيمكن استخدام الدالة setLayout لتحديد نوع التخطيط الخاص باللوحة المعينة .

* تم تقسيم الاطار اعلاه الي لوحتين الاولي للازرار والثانية لاحتواء مربع النص الاول واللوحة الاولي ..... ويتم وضع مربع النص الثاني في منتصف الاطار.

* الاسطر 21-22 تقوم باضافة مربع ادخال لاستقبال القيم من المستخدم واضافتها الي اللوحة الثانية

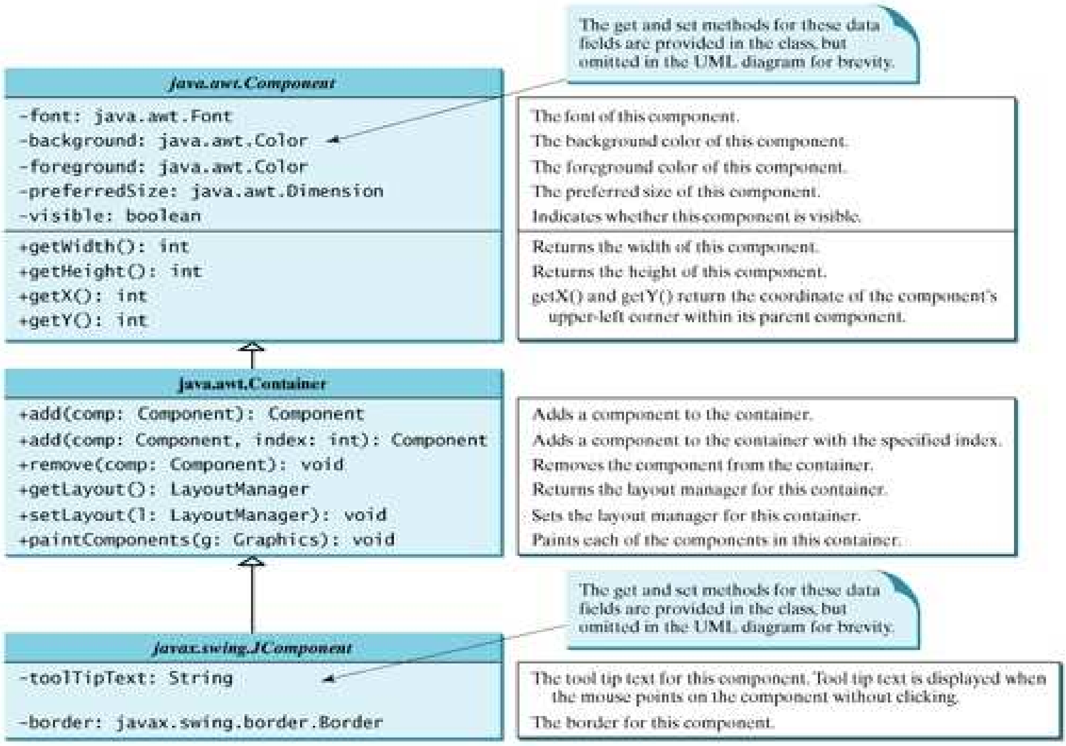
ملحوظة مهمة :-

الصنف Contianer هو الصنف الاب لكل اصناف *.javax.swing وكل الدوال الموجودة فيه يتم توريثها الي الاصناف الفرعية منها .

لذلك (نظريا) يمكن استخدام الازرار كحاويات فرعية ولكن لاسباب (عملية) لا يفضل استخدام الازرار لهذا الغرض.

خصائص مشتركة بين مكونات الواجهات الرسومية في الجافا
Common Features of Swing GUI Components

* لقد استخدمنا عدد من مكونات الواجهات الرسومية في الجافا مثل Jframe , Jpanel , Jbutton , JtextField , Jlabel ... وسنقوم بالتعرف علي مكونات اكثر في الفصول القادمة ولكن من المهم اولا فهم الخصائص المشتركة بين مكونات الواجهات الرسومية .
* صنف Component هو الصنف الاساسي لكل اصناف الواجهات الرسومية و الحاويات وكل اصناف swing .
* كل اصناف swing ماعدا (Jframe , Container , Jdailog) عبارة عن اصناف فرعية من الصنف Jcomponent
* الشكل التالي يعرض عدد من الدوال الاكثر استخداما في الواجهات الرسومية للتحكم في الخصائص مثل الحجم و الخط واللون و .... .



* الدالة toolTipText تقوم بعرض نص توضيحي عن عمل المكون الرسومي عن الاشارة اليه بواسطة الماوس.
* يمكن انشاء حدود للمكونات عن طريق دوال التهيئة التالية :

```
new TitledBorder(String title);
new LineBorder(Color color, int width);
```

مثال :-

توضيح لاستخدام الخصائص في واجهة رسومية ؟؟؟

الحل

شكل الواجهة المطلوب برمجتها باستخدام نفس الخصائص

الكوووود :-

```java
import java.awt.*;
import javax.swing.*;
import javax.swing.border.*;

public class TestSwingCommonFeatures extends JFrame {
  public TestSwingCommonFeatures() {
    // Create a panel to group three buttons
    JPanel p1 = new JPanel(new FlowLayout(FlowLayout.LEFT, 2, 2));
    JButton jbtLeft = new JButton("Left");
    JButton jbtCenter = new JButton("Center");
    JButton jbtRight = new JButton("Right");
    jbtLeft.setBackground(Color.WHITE);
    jbtCenter.setForeground(Color.GREEN);
    jbtRight.setToolTipText("This is the Right button");
    p1.add(jbtLeft);
    p1.add(jbtCenter);
    p1.add(jbtRight);
    p1.setBorder(new TitledBorder("Three Buttons"));

    // Create a font and a line border
    Font largeFont = new Font("TimesRoman", Font.BOLD, 20);
    Border lineBorder = new LineBorder(Color.BLACK, 2);

    // Create a panel to group two labels
    JPanel p2 = new JPanel(new GridLayout(1, 2, 5, 5));
    JLabel jlblRed = new JLabel("Red");
    JLabel jlblOrange = new JLabel("Orange");
    jlblRed.setForeground(Color.RED);
    jlblOrange.setForeground(Color.ORANGE);
    jlblRed.setFont(largeFont);
    jlblOrange.setFont(largeFont);
    jlblRed.setBorder(lineBorder);
    jlblOrange.setBorder(lineBorder);
    p2.add(jlblRed);
    p2.add(jlblOrange);
    p2.setBorder(new TitledBorder("Two Labels"));

    // Add two panels to the frame
    setLayout(new GridLayout(2, 1, 5, 5));
    add(p1);
    add(p2);
  }

  public static void main(String[] args) {
```

```java
        // Create a frame and set its properties
        JFrame frame = new TestSwingCommonFeatures();
        frame.setTitle("TestSwingCommonFeatures");
        frame.setSize(300, 150);
        frame.setLocationRelativeTo(null); // Center the frame
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

* ملحوظة مهمة :-

نفس الخاصية يمكن ان يكون لها قيم مختلفة في اماكن مختلفة ، مثلا الخاصية Visible قيمتها الافتراضية في Jframe هي false ولذلك لاظهار الاطار لابد من تحويلها الي true.

اما في حالة باقي المكونات مثل Jbutton تكون قيمة Visible الافتراضية true فلا داعي لتحويلها الي الي قيمة اخري .

الصور

Image Icons

* في الجافا يتم استخدام الصنف javax.swing.ImageIcon لتمثيل الايقونات في الواجهات ،.

* الايقونة هي عبارة عن صورة ذات حجم محدد وغالبا هي صغيرة الابعاد وتستخدم لتزيين المكونات الاخري .

* نستخدم دالة التهيئة التالية لانشاء ايقونة صورة :

ImageIcon icon = new ImageIcon(filename);

حيث ان filename عبارة عن مسار الصورة التي نريد تحويلها الي ايقونة .

ملحوظة مهمة :-
تدعم الجافا 3 انواع من الصور GIF , JPEG , PNG واسماء الصور من هذه الثلاثة انواع تنتهي بالاختصارات التالية علي الترتيب (.png , .jpeg , .gif) . واذا كانت الصورة التي تريد وضعها من نوع غير هذه الانواع فيمكن استخدام برنامج الرسام لتحويلها الي هذه الانواع.

مثال :

برنامج يعرض 4 ايقونات صور ؟؟

الحل

الكوود

```java
import javax.swing.*;
import java.awt.*;

public class TestImageIcon extends JFrame {
  private ImageIcon usIcon = new ImageIcon("image/us.gif");
  private ImageIcon myIcon = new ImageIcon("image/my.jpg");
```

[Page 419]

```java
  private ImageIcon frIcon = new ImageIcon("image/fr.gif");
  private ImageIcon ukIcon = new ImageIcon("image/uk.gif");

  public TestImageIcon() {
    setLayout(new GridLayout(1, 4, 5, 5));
```

```java
    add(new JLabel(usIcon));
    add(new JLabel(myIcon));
    add(new JButton(frIcon));
    add(new JButton(ukIcon));
  }

  /** Main method */
  public static void main(String[] args) {
    TestImageIcon frame = new TestImageIcon();
    frame.setTitle("TestImageIcon");
    frame.setLocationRelativeTo(null); // Center the frame
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(200, 200);
    frame.setVisible(true);
  }
}
```

ملحوظة مهمة :-

لا يمكن مشاركة احد المكونات بين عدد من اللوحات نسبة لان المكون يظهر في لوحة واااحدة فقط ،، والعلاقة بين المكون والحاوية علاقة انتماء.

يمكن مشاركة الحدود و الايقونات بين المكونات المختلفة .

Chapter Summary

ملخص الفصل الاول

* كل حاوية تحتوي علي مدير تخطيط لتنظيم طريقة ترتيب المكونات في الحاوية حسب الاماكن المرغوبة ، ومن امثلة المدراء البسيطيين والسهلين FlowLayout , GridLayout , BorderLayout .

* يمكن استخدام Jpanel كحاوية فرعية للحصول علي الشكل المطلوب.

* قم باستخدام الدالة add لاضافة الJPanel الي الاطار ، وافتراضيا ان التخطيط الخاص باللوحة هو FlowLayout ،، وللاطار BorderLayout .

* يمكن التحكم في الوان مكونات الواجهة عن طريق الصنف java.awt.Color عادة تتكون الالوان من مزج ثلاثة الوان خي الاحمر والاخضر والازرق وكل منها يمثل برقم من دون اشارة يترواح بين 0 و 255 وهذا الظام هو ما يعرف RGB Model .

Review Questions

Sections 12.3–12.4

12.1 Which class is the root of the Java GUI component classes? Is a container class a subclass of `Component`? Which class is the root of the Swing GUI component classes?

12.2 Explain the difference between AWT GUI components, such as `java.awt.Button`, and Swing components, such as `javax.swing.JButton`.

12.3 How do you create a frame? How do you set the size for a frame? How do you get the size of a frame? How do you add components to a frame? What would happen if the statements `frame.setSize(400, 300)` and `frame.setVisible(true)` were swapped in the `MyFrameWithComponents` class in Section 12.4.2, "Adding Components to a Frame"?

12.4 Determine whether the following statements are true or false:

- You can add a button to a frame.
- You can add a frame to a panel.
- You can add a panel to a frame.
- You can add any number of components to a panel or a frame.
- You can derive a class from `JButton`, `JPanel`, `JFrame`, or `JApplet`.

12.5 The following program is supposed to display a button in a frame, but nothing is displayed. What is the problem?

```
public class Test extends javax.swing.JFrame {
  public Test() {
    add(new javax.swing.JButton("OK"));
  }

  public static void main(String[] args) {
    javax.swing.JFrame frame = new javax.swing.JFrame();
    frame.setSize(100, 200);
    frame.setVisible(true);
  }
}
```

12.6 Which of the following statements have syntax errors?

```
Component c1 = new Component();
```

```
JComponent c2 = new JComponent();
Component c3 = new JButton();
JComponent c4 = new JButton();
Container c5 = new JButton();
c5.add(c4);
Object c6 = new JButton();
c5.add(c6);
```

Section 12.5 Layout Managers

12.7 Why do you need to use layout managers? What is the default layout manager for a frame? How do you add a component to a frame?

12.8 Describe `FlowLayout`. How do you create a `FlowLayout` manager? How do you add a component to a `FlowLayout` container? Is there a limit to the number of components that can be added to a `FlowLayout` container?

12.9 Describe `GridLayout`. How do you create a `GridLayout` manager? How do you add a component to a `GridLayout` container? Is there a limit to the number of components that can be added to a `GridLayout` container?

12.10 Describe `BorderLayout`. How do you create a `BorderLayout` manager? How do you add a component to a `BorderLayout` container? Can you add multiple components in the same section?

Sections 12.6–12.7

12.11 How do you create a color? What is wrong about creating a `Color` using `new Color(400, 200, 300)`? Which of the following two colors are darker, `new Color(10, 0, 0)` or `new Color(200, 0, 0)`?

12.12 How do you create a font? How do you find all the available fonts on your system?

Section 12.8 Using Panels as Subcontainers

12.13 How do you create a panel with a specified layout manager?

12.14 What is the default layout manager for a `JPanel`? How do you add a component to a `JPanel`?

12.15 Can you use the `setTitle` method in a panel? What is the purpose of using a panel?

12.16 Since a GUI component class such as `JButton` is a subclass of `Container`, can you add components into a button?

Sections 12.9–12.10

12.17 How do you set background color, foreground color, font, and tool tip text on a Swing GUI component? Why is the tool tip text not displayed in the following code?

```
import javax.swing.*;

public class Test extends JFrame {
  private JButton jbtOK = new JButton("OK");

  public static void main(String[] args) {
    // Create a frame and set its properties
    JFrame frame = new Test();
    frame.setTitle("Logic Error");
    frame.setSize(200, 100);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
  }

  public Test() {
    jbtOK.setToolTipText("This is a button");
    add(new JButton("OK"));
  }
}
```

12.18 Show the output of the following code:

```
import javax.swing.*;

public class Test {
  public static void main(String[] args) {
    JButton jbtOK = new JButton("OK");
    System.out.println(jbtOK.isVisible());

    JFrame frame = new JFrame();
    System.out.println(frame.isVisible());
  }
}
```

12.19 How do you create an `ImageIcon` from the file image/us.gif in the class directory?

12.20 What happens if you add a button to a container several times, as shown below? Does it cause syntax errors? Does it cause runtime errors?

```
JButton jbt = new JButton();
JPanel panel = new JPanel();
panel.add(jbt);
panel.add(jbt);
```

```
panel.add(jbt);
```

12.21 Will the following code display three buttons? Will the buttons display the same icon?

```
import javax.swing.*;
import java.awt.*;

public class Test extends JFrame {
  public static void main(String[] args) {
    // Create a frame and set its properties
    JFrame frame = new Test();
    frame.setTitle("ButtonIcons");
    frame.setSize(200, 100);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
  }

  public Test() {
    ImageIcon usIcon = new ImageIcon("image/usIcon.gif");
    JButton jbt1 = new JButton(usIcon);
    JButton jbt2 = new JButton(usIcon);

    JPanel p1 = new JPanel();
    p1.add(jbt1);


    JPanel p2 = new JPanel();
    p2.add(jbt2);

    JPanel p3 = new JPanel();
    p2.add(jbt1);

    add(p1, BorderLayout.NORTH);
    add(p2, BorderLayout.SOUTH);
    add(p3, BorderLayout.CENTER);
  }
}
```

12.22 Can a border or an icon be shared by GUI components?

انتهى الفصل الاول بحمد الله .....